

LAB 7

MỤC TIÊU

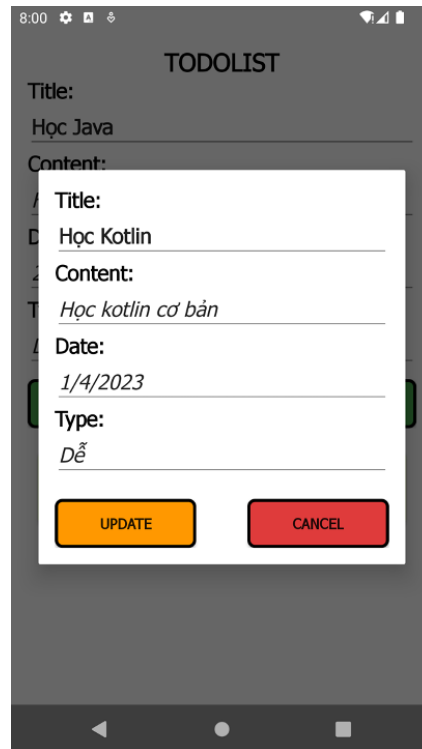
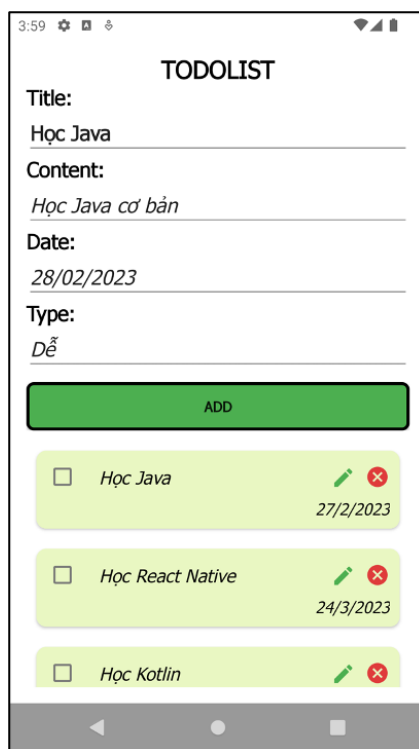
Kết thúc bài thực hành sinh viên có khả năng:

- ✓ Biết cách setup một project Firebase Cloud Firestore
- ✓ CRUD của Firebase Cloud Firestore.

NỘI DUNG :

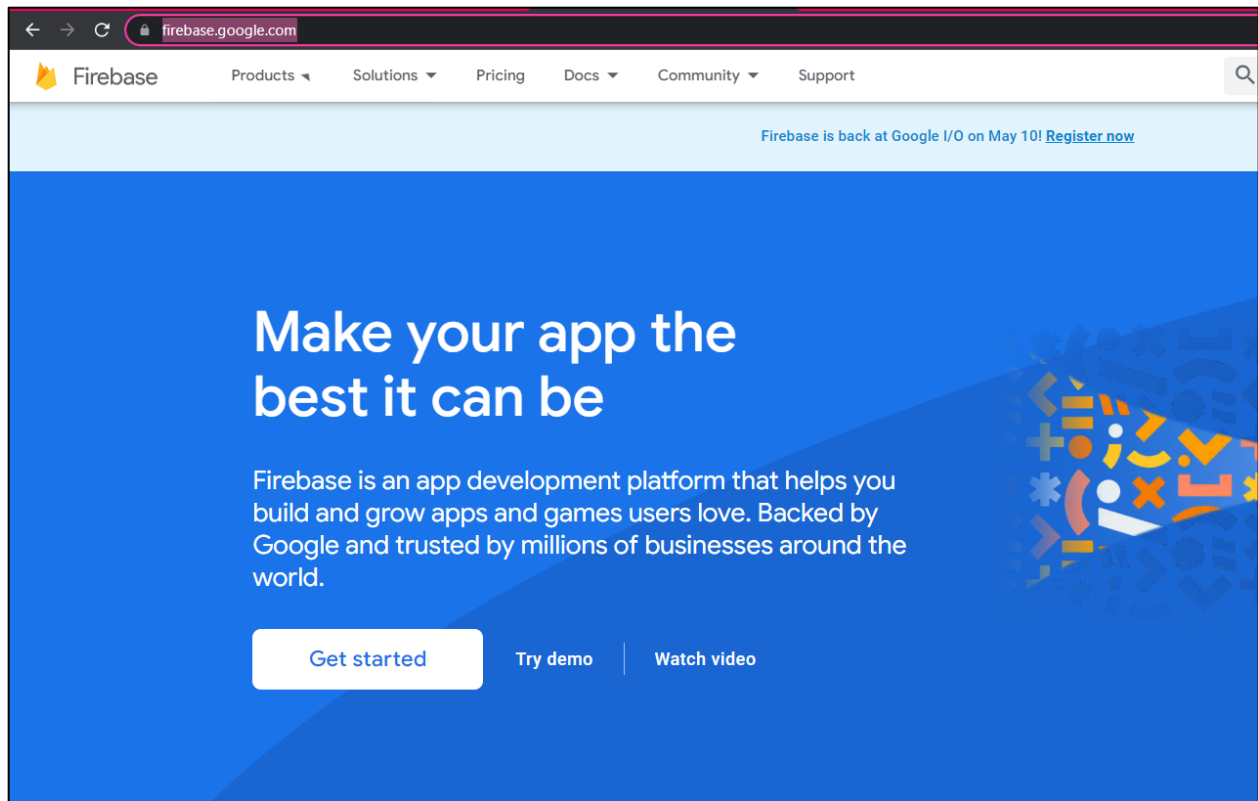
Tạo một ứng dụng **TODOLIST** với database bằng **Firebase Cloud Firestore**

Sử dụng lại các layout từ Project Lab1, Lab2, Lab3 để thực hiện

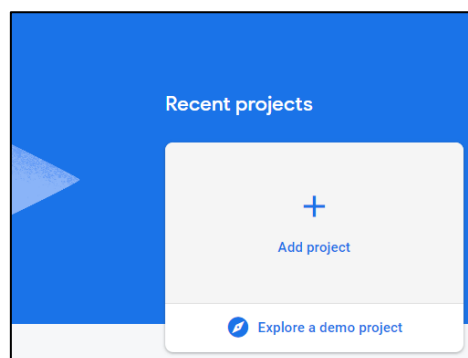


BÀI 1: Setup Project Firebase Cloud Firestore

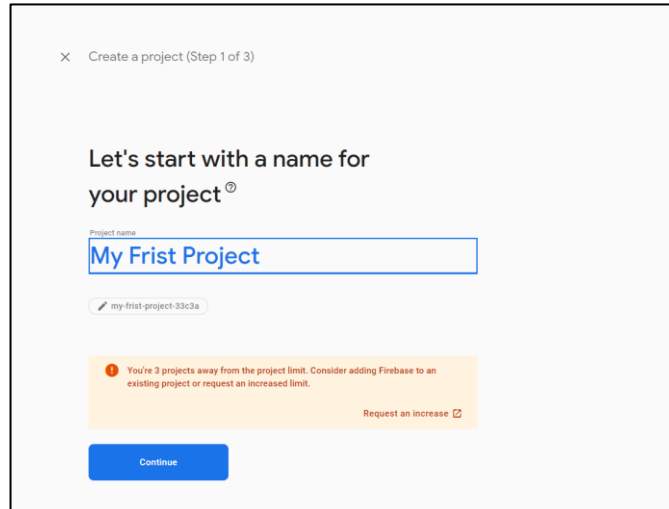
Bước 1: Truy cập vào <https://firebase.google.com/> đăng nhập tài khoản. Sau đó chọn **Get started**



Chọn **Add project**



Đặt tên project và nhấn **Continue**



X Create a project (Step 1 of 3)

Let's start with a name for your project[®]

Project name

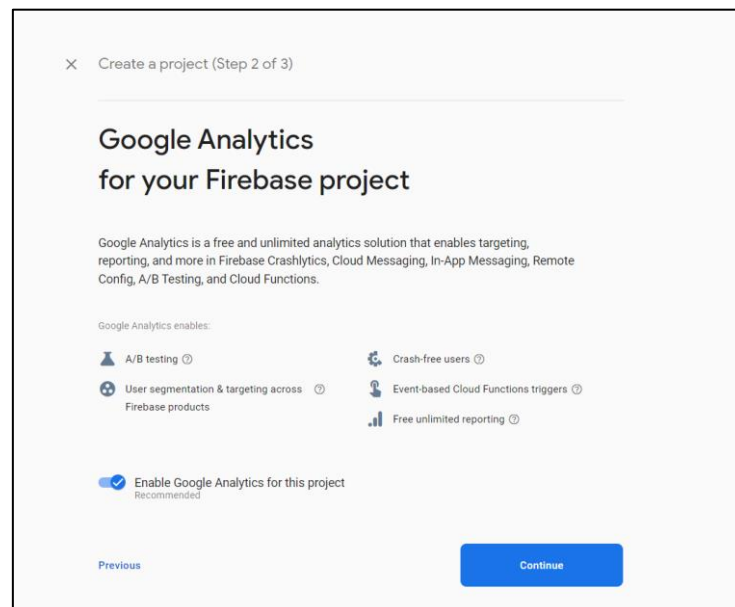
My Frist Project

my-frist-project-33c3a

You're 3 projects away from the project limit. Consider adding Firebase to an existing project or request an increased limit. [Request an increase](#)

Continue

Tiếp tục chọn **Continue**



X Create a project (Step 2 of 3)

Google Analytics
for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

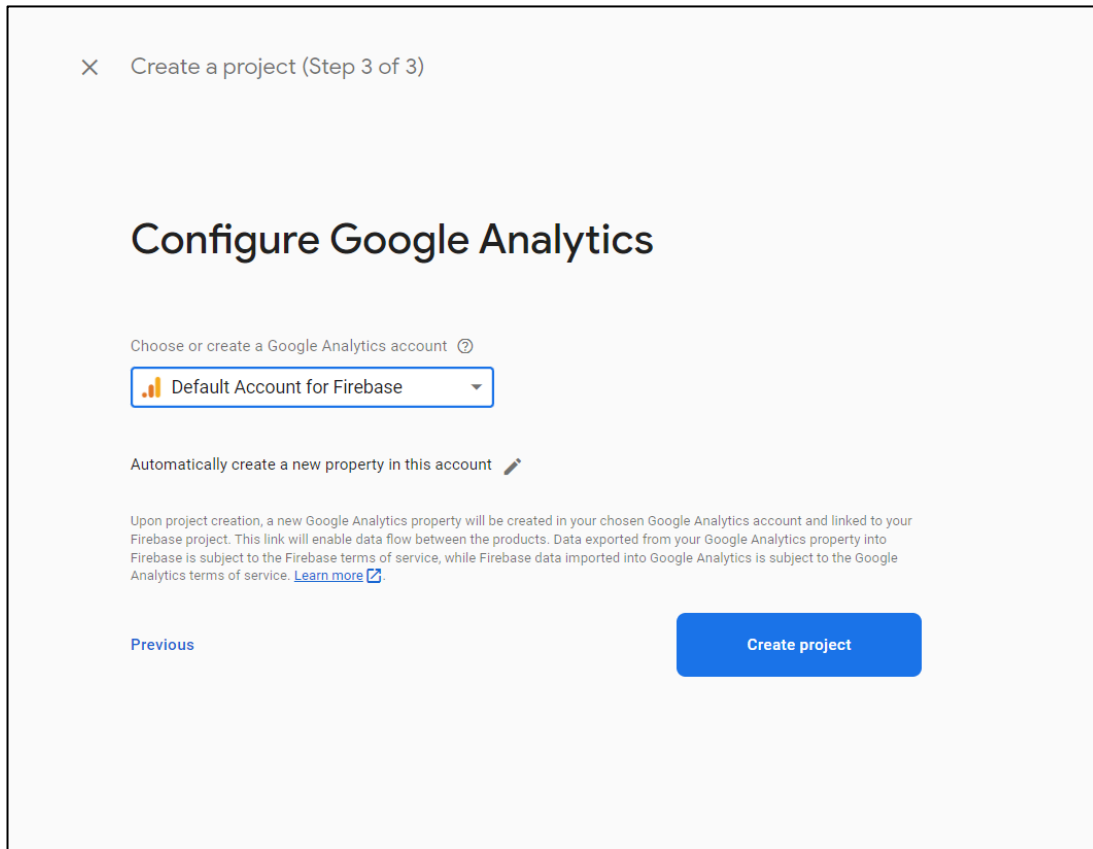
Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

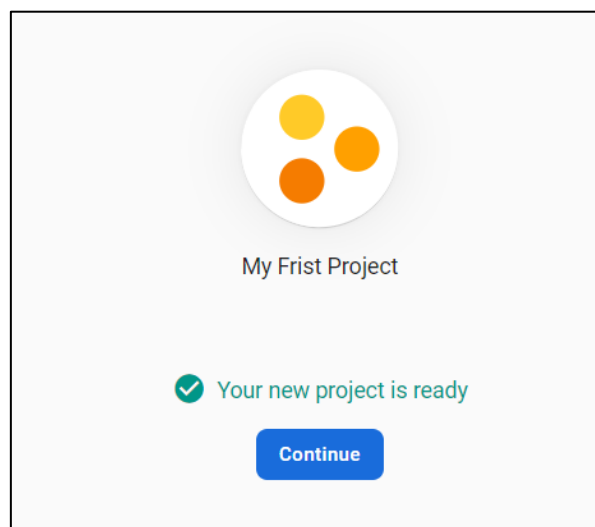
☒ Enable Google Analytics for this project
Recommended

Previous Continue

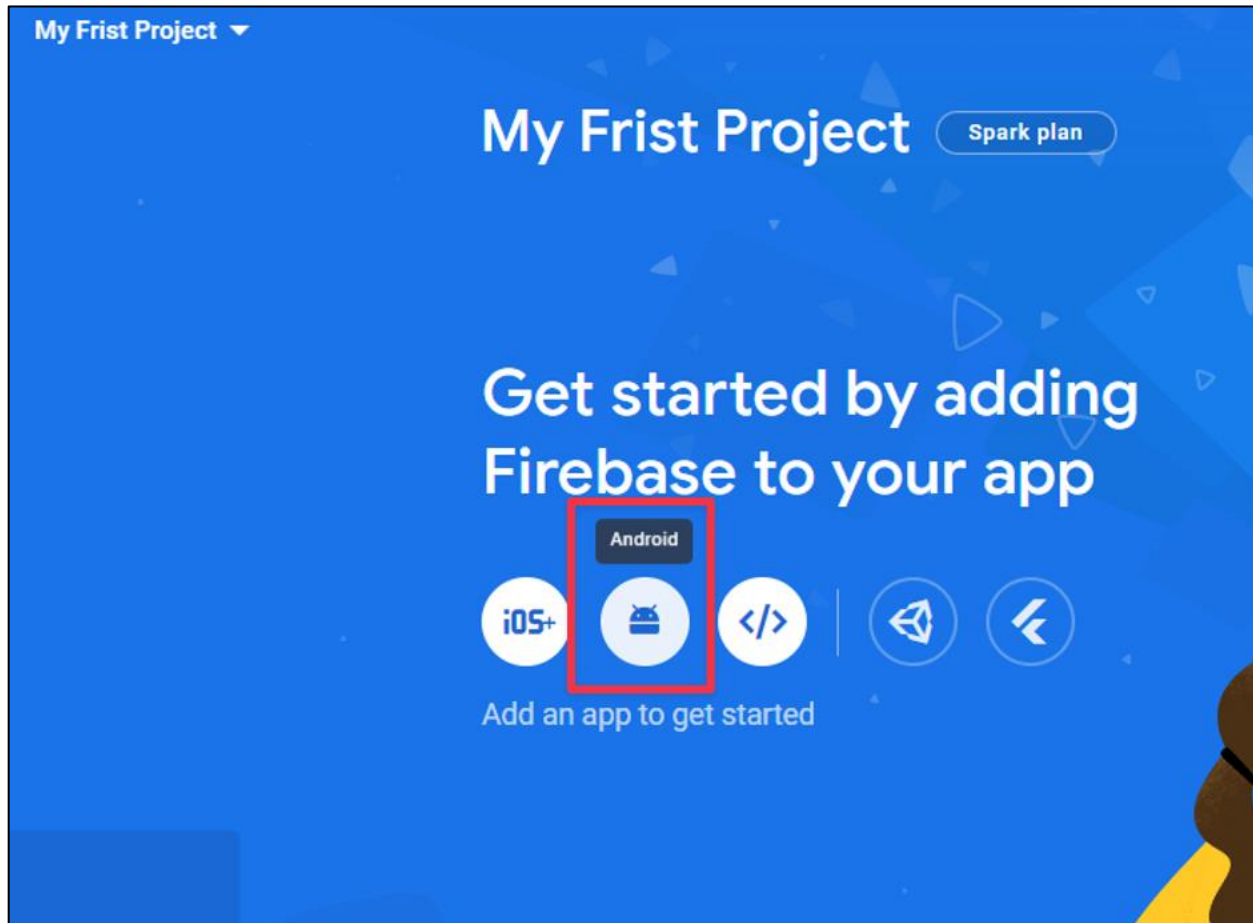
Chọn **Default Account for Firebase** và nhấn **Create Project**



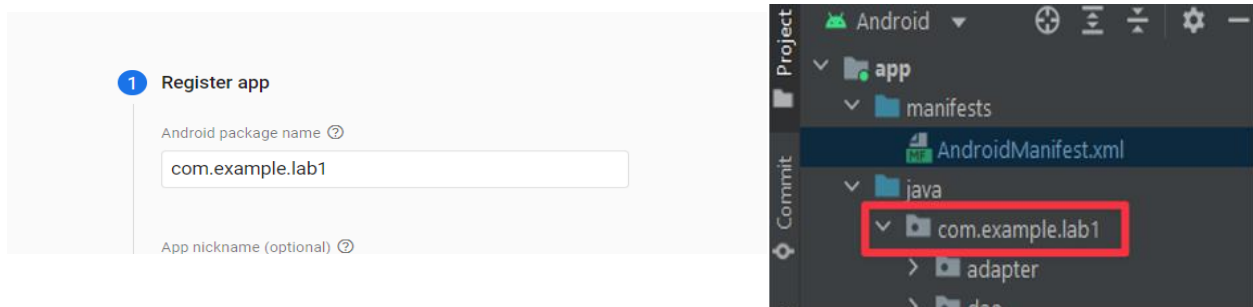
Sau khi xong khi Firebase tạo xong Project, ta chọn **Continue** để bắt đầu sử dụng



Chọn vào biểu tượng Android để bắt đầu setup



- Điền tên package của app



- App nickname đặt tên tùy ý

App nickname (optional) ?

Cloud Firestore

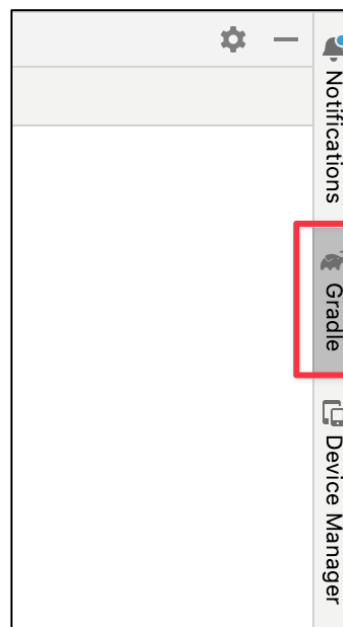
- Điền thông tin mã **SHA-1**, chúng ta sẽ lấy được mã này trong Android Studio

Debug signing certificate SHA-1 (optional) ?

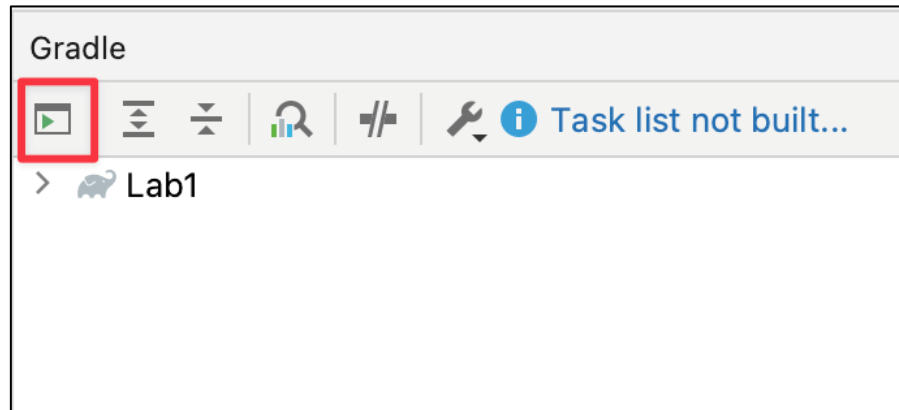
|00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:

i Required for Dynamic Links, and Google Sign-In or phone number support in Auth.
Edit SHA-1s in Settings.

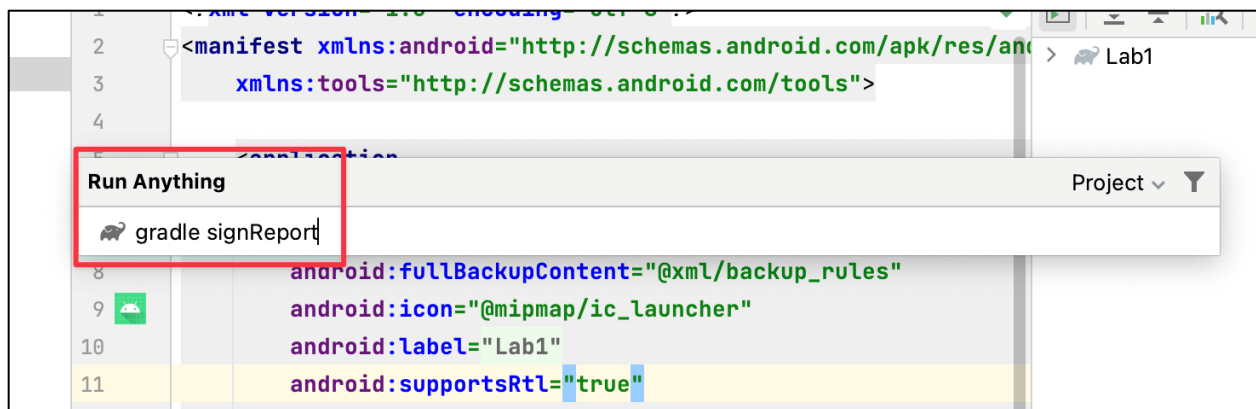
Trong Android Studio, ở góc phải màn hình chọn và tag **Gradle**



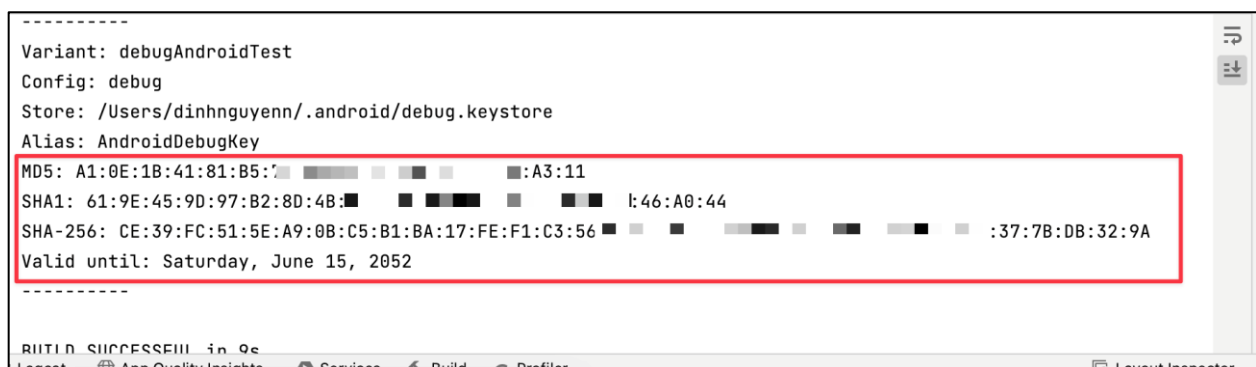
Chọn biểu tượng **Execute Gradle Task**



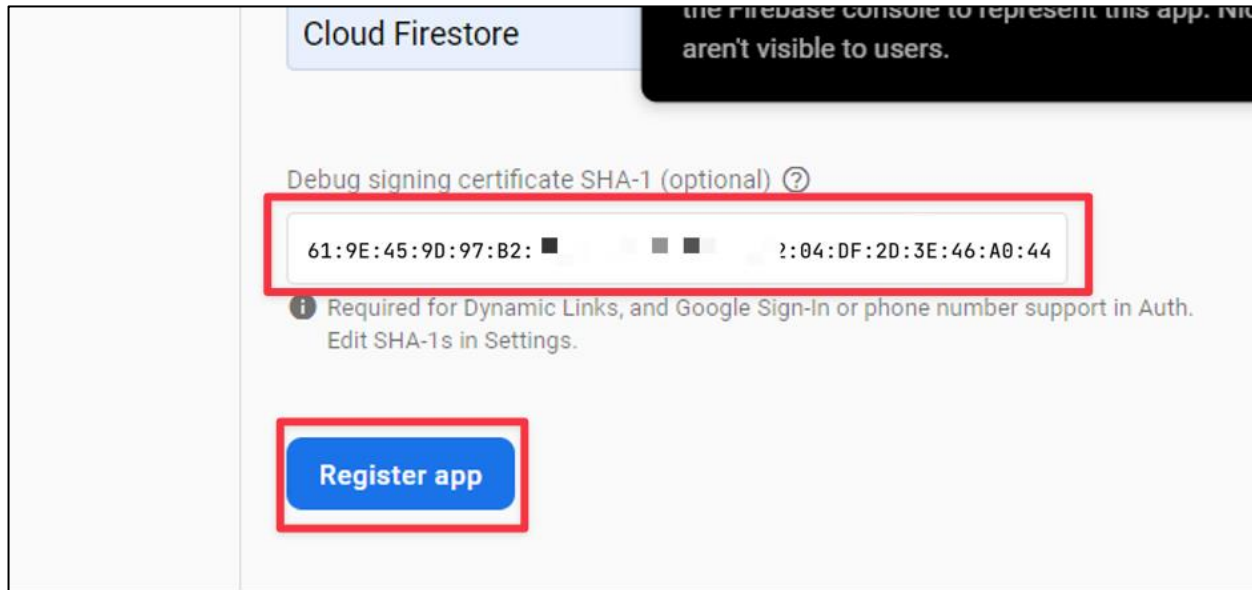
Gõ vào ô tìm kiếm **signReport** và nhấn enter



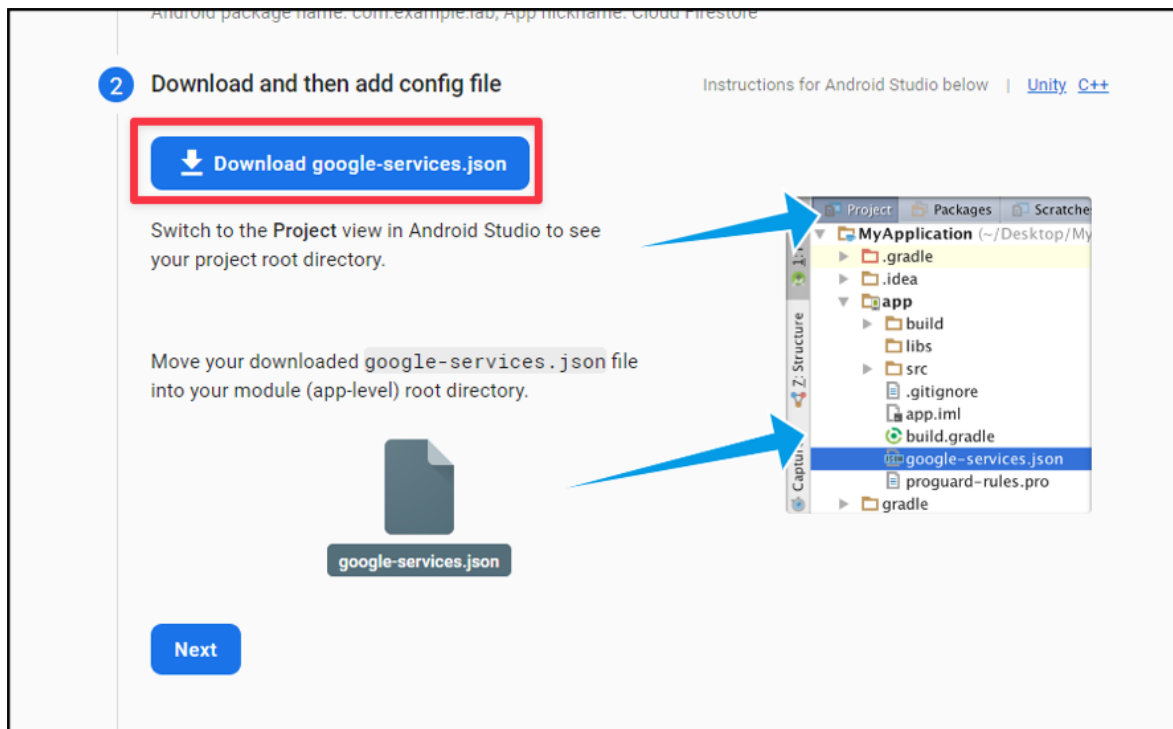
Sau đó sẽ có cửa sổ Terminal hiện lên copy SHA-1



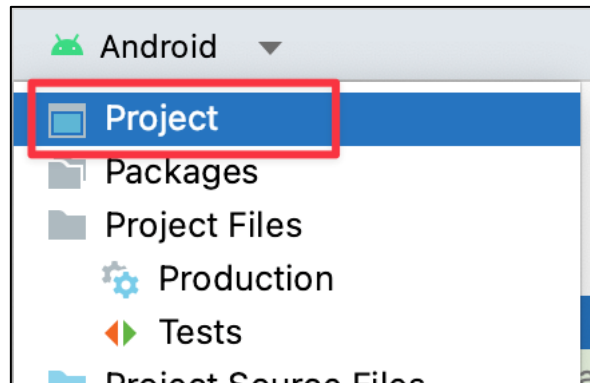
Paste SHA-1 vào ô như hình bên dưới, sau đó nhấn tiếp **Register app**



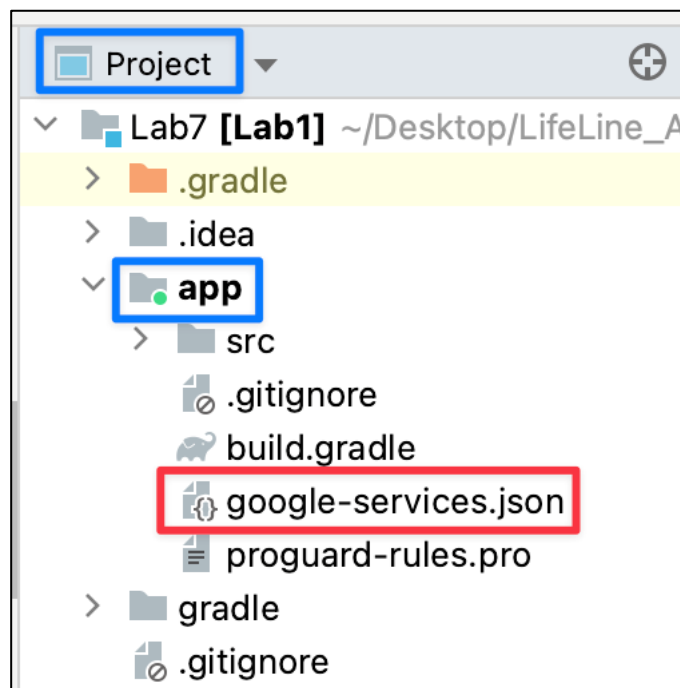
Bước 2: Click vào nút “**Download google-services.json**” để tải file config về máy



Paste file config vừa tải vào project Android bằng cách thay đổi cấu trúc view của project từ **Android** sang **Project**



Copy file config vừa tải vào thư mục **app**



Bước 3: Thêm các thông số, thư viện cần thiết

3 Add Firebase SDK

Instructions for Gradle | [Unity](#) [C++](#)

1. To make the `google-services.json` config values accessible to Firebase SDKs, you need the Google services Gradle plugin.

Add the plugin as a buildscript dependency to your project-level `build.gradle` file:

Root-level (project-level) Gradle file (`<project>/build.gradle`):

```

buildscript {
    repositories {
        // Make sure that you have the following two repositories
        google() // Google's Maven repository
        mavenCentral() // Maven Central repository
    }
    dependencies {
        ...
        // Add the dependency for the Google services Gradle plugin
        classpath 'com.google.gms:google-services:4.3.15'
    }
}

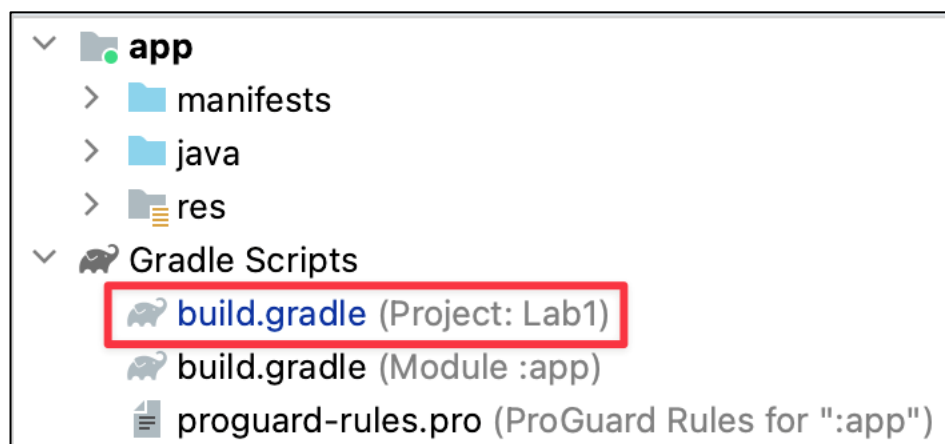
allprojects {
    ...
    repositories {
        // Make sure that you have the following two repositories
        google() // Google's Maven repository
        mavenCentral() // Maven Central repository
    }
}

```

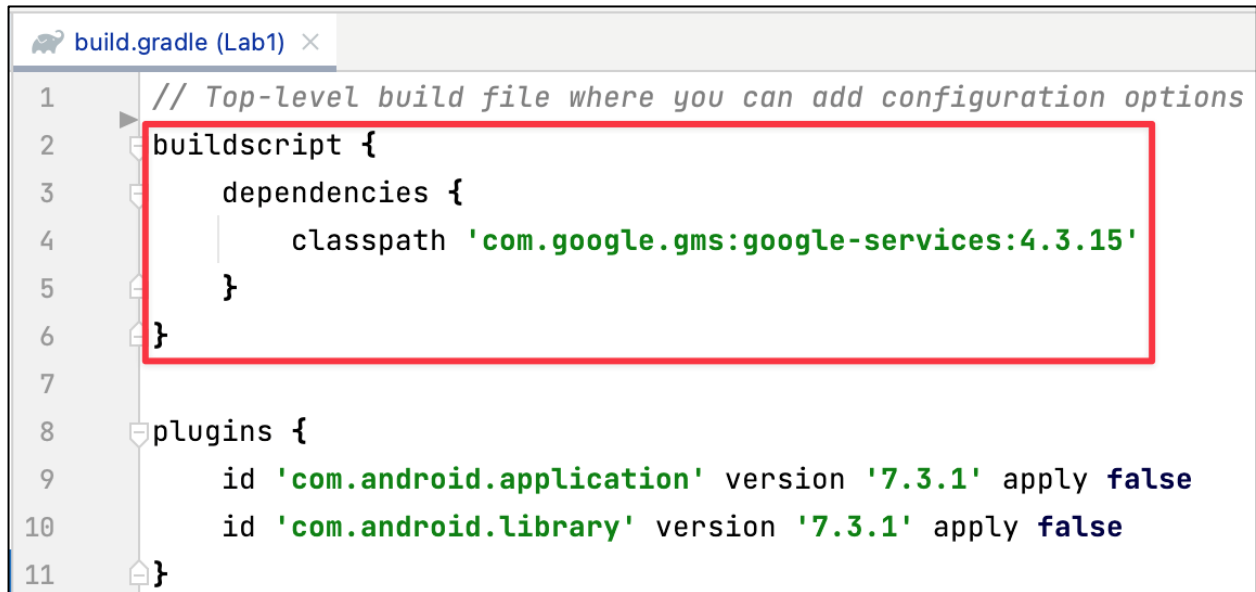
2. Then, in your module (app-level) `build.gradle` file, add both the `google-services` plugin and any Firebase SDKs that you want to use in your app:

☐ Kotlin
 ☒ Java

Trong `build.gradle` (Module: Project)

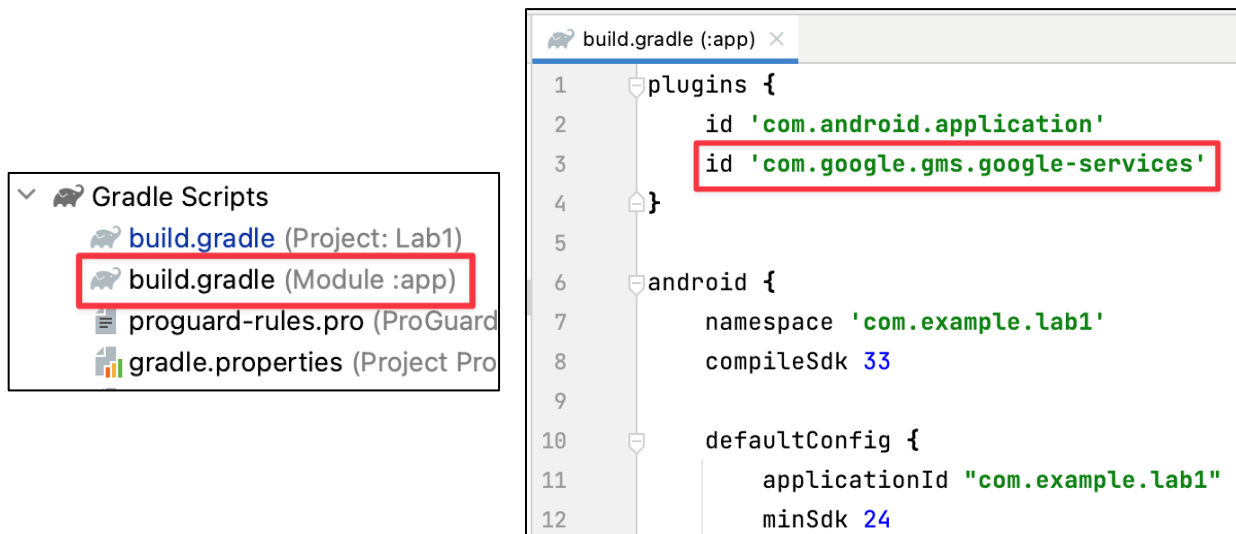


Thêm đoạn code sau (copy ở bước 3 trên Firebase)



```
1 // Top-level build file where you can add configuration options
2 buildscript {
3     dependencies {
4         classpath 'com.google.gms:google-services:4.3.15'
5     }
6 }
7
8 plugins {
9     id 'com.android.application' version '7.3.1' apply false
10    id 'com.android.library' version '7.3.1' apply false
11 }
```

Tiếp theo trong file **build.gradle (Module)**, phần plugins thêm đoạn code:

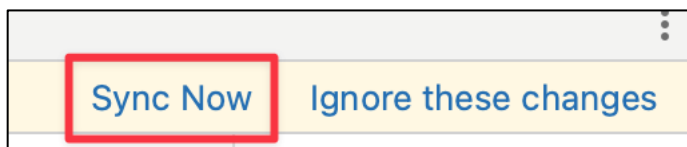


```
1 plugins {
2     id 'com.android.application'
3     id 'com.google.gms.google-services'
4 }
5
6 android {
7     namespace 'com.example.lab1'
8     compileSdk 33
9
10    defaultConfig {
11        applicationId "com.example.lab1"
12        minSdk 24
13    }
14 }
```

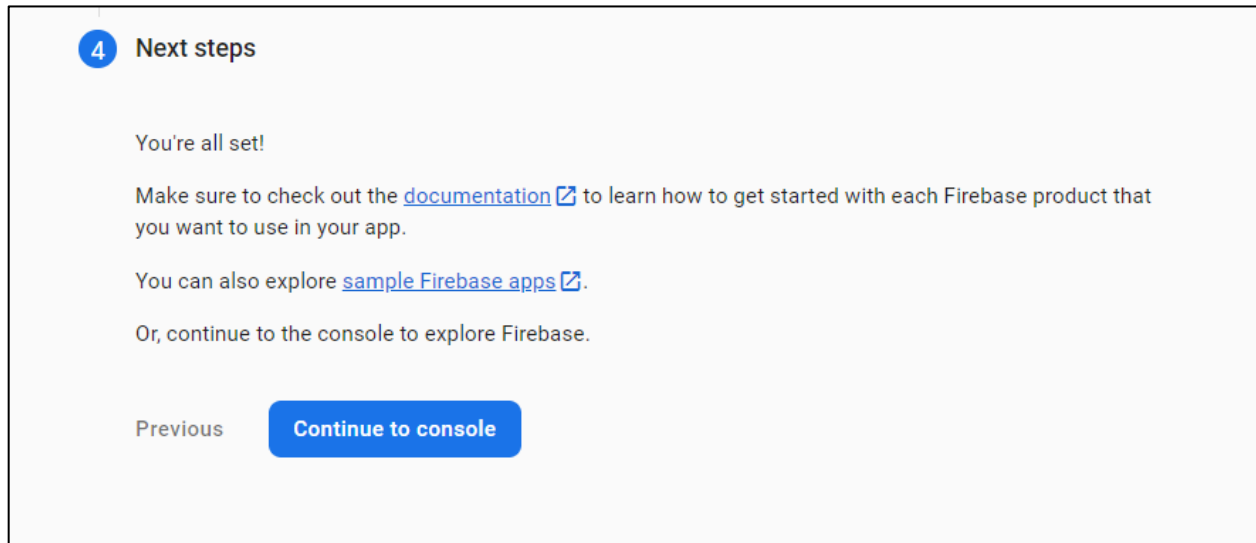
Tại **dependencies** trong **build.gradle (Moule)** thêm thư viện:

```
dependencies {  
  
    implementation 'androidx.appcompat:appcompat:1.6.1'  
    implementation 'com.google.android.material:material:1.8.0'  
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'  
    testImplementation 'junit:junit:4.13.2'  
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'  
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'  
  
    //Firebase  
    implementation platform('com.google.firebase:firebase-bom:31.2.3')  
    implementation 'com.google.firebase:firebase-analytics'  
    //Cloud Firestore  
    implementation 'com.google.firebase:firebase-firestore'  
  
}
```

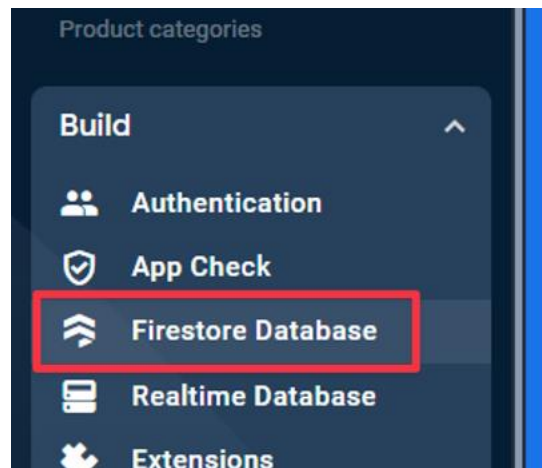
Sau đó nhấn **Sync Now**



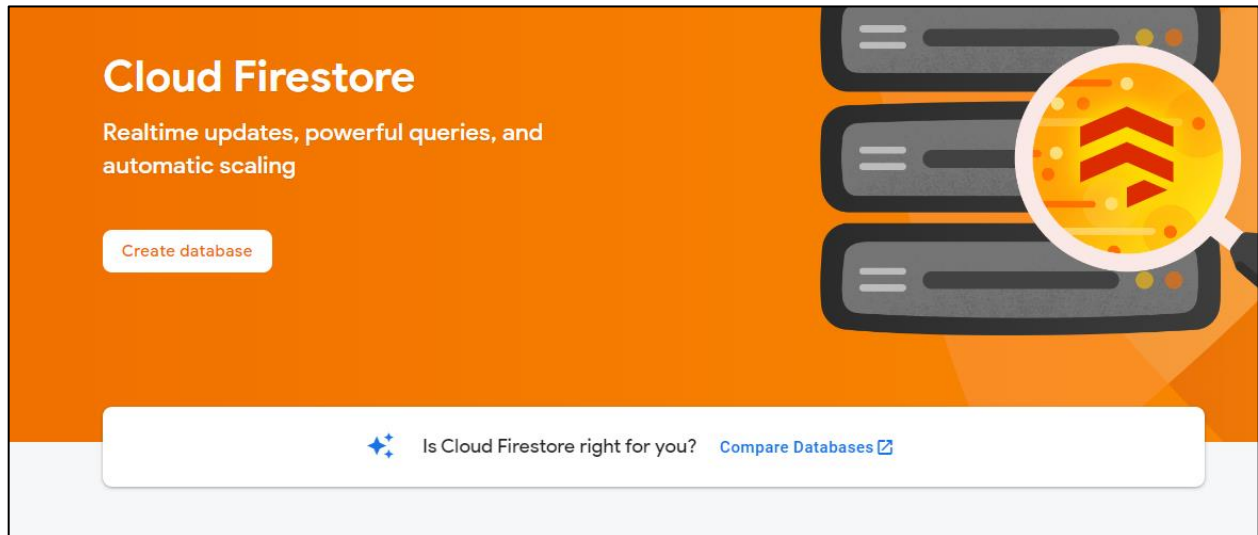
Trên Firebase nhấn **Next** và tiếp tục nhấn **Continue to console**



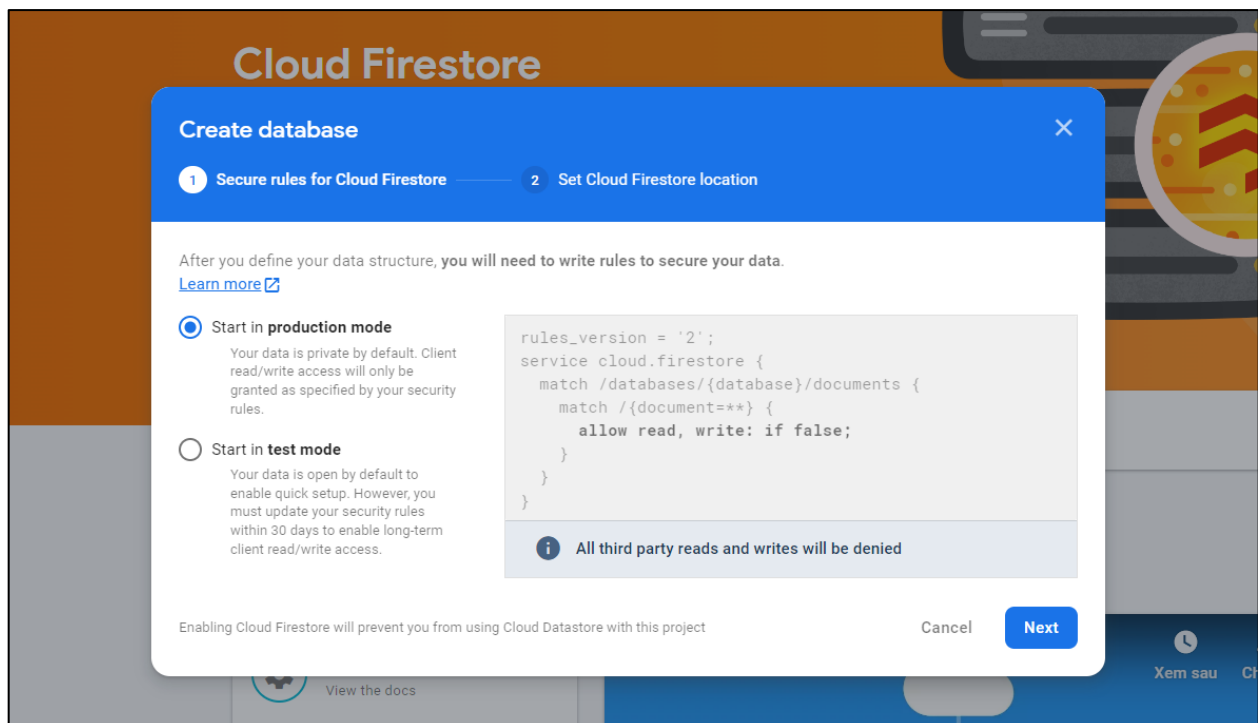
Bước 4: Ở giao diện **Console Firebase** trong mục **Build** chọn **Firestore Database**



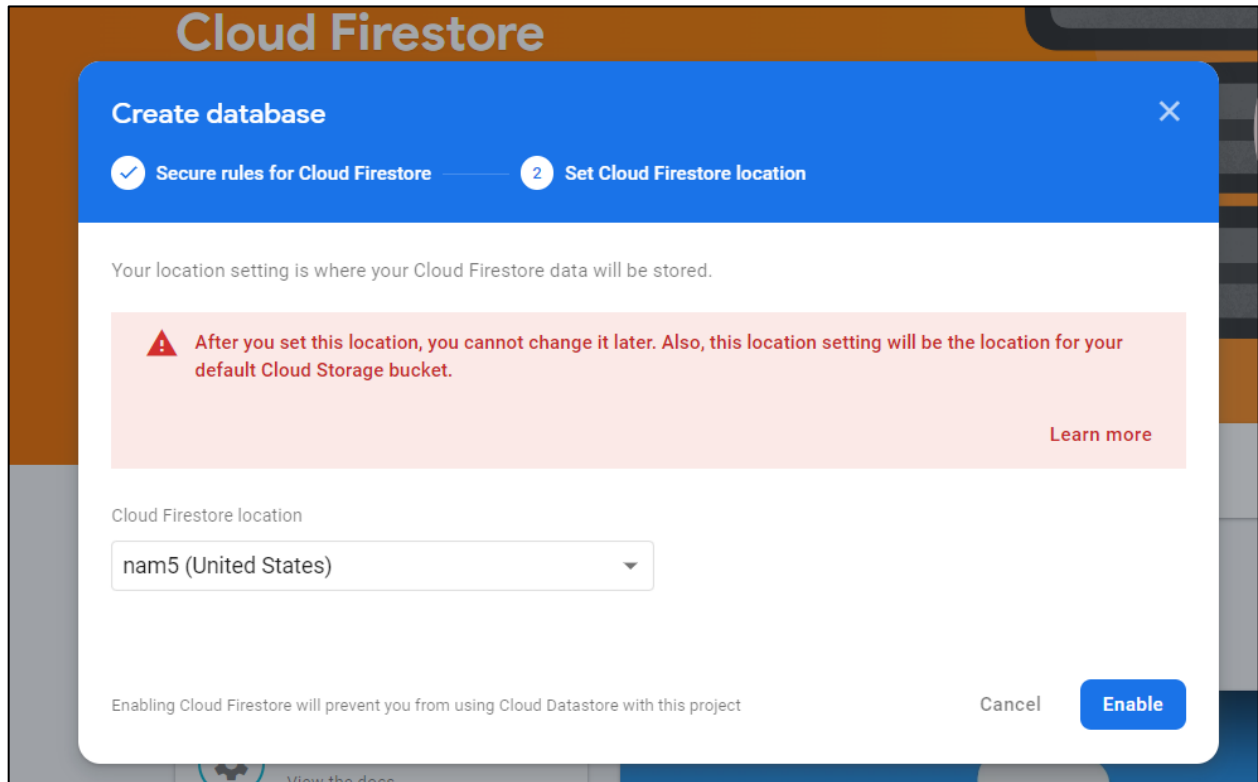
Sau đó chọn **Create database**



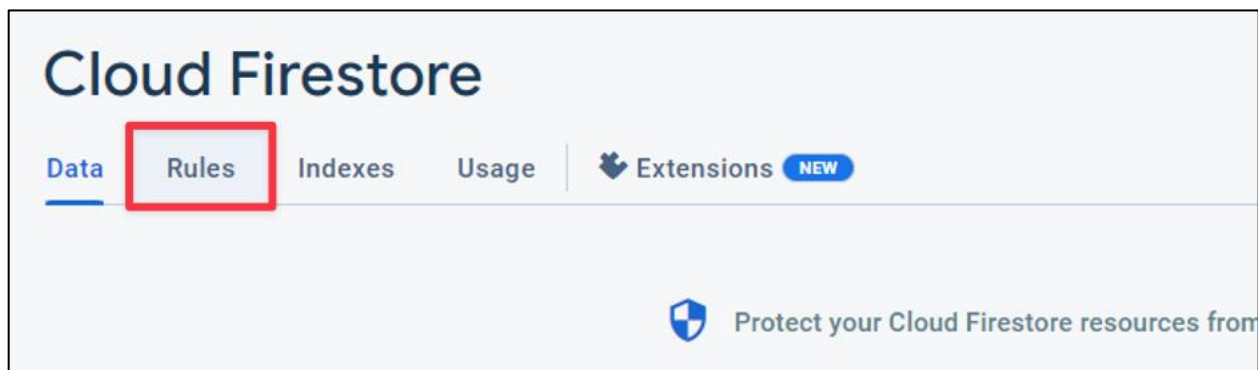
Chọn **production mode** rồi nhấn **Next**



Nhấn chọn location sau đó nhấn **Enable**



Sau đó tại Firestore bạn chọn qua tab **Rules**



Sửa lại nội dung như sau:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if true;
    }
  }
}
```

Ta đã cấu hình xong project firebase với **Firestore database**

BÀI 2: Thực hiện chức năng thêm một công việc mới ở Lab 1, 2, 3 bằng cách sử dụng Firestore Database

Bước 1: Với mong muốn tạo **id** công việc một cách ngẫu nhiên và tự động, ta thay đổi kiểu dữ liệu của thuộc tính **id** trong model Todo từ **int** thành **String**

```
public class Todo {  
    4 usages  
    private String id;  
    4 usages  
    private String title, content, date, type;  
    4 usages  
    private int status;  
}
```

Thay đổi **constructor**, **get**, **set** của model **Todo**, kiểm tra và chỉnh sửa một số đoạn code cho phù hợp sau khi thay đổi kiểu dữ liệu của thuộc tính **id**

Ta sẽ thêm một hàm mới đặt tên là **convertHashMap** trong model **Todo**, hàm này có nhiệm vụ xử lý dữ liệu thao tác trên Firestore Database

```
public HashMap<String, Object> convertHashMap() {  
    HashMap<String, Object> work = new HashMap<>();  
    work.put("id", id);  
    work.put("title", title);  
    work.put("content", content);  
    work.put("date", date);  
    work.put("type", type);  
    work.put("status", status);  
    return work;  
}
```

Bước 2: Kết nối với Firebase và tạo tên cho Collection

Tạo biến toàn cục FirebaseFirestore

```
//Tạo biến toàn cục FirebaseFirestore  
7 usages  
FirebaseFirestore database;
```

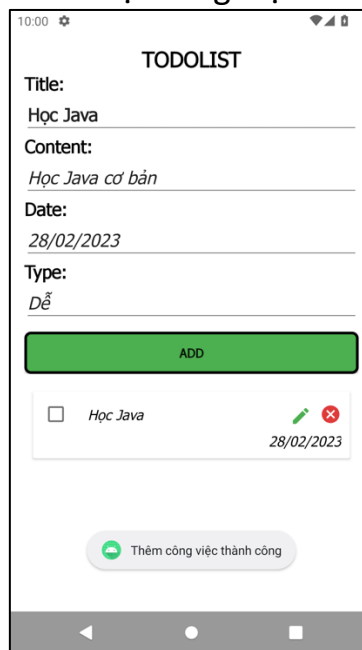
Trong hàm **onCreate()** kết nối với FirebaseFirestore đã tạo trước đó

```
//Kết nối với database hiện tại  
database = FirebaseFirestore.getInstance();
```

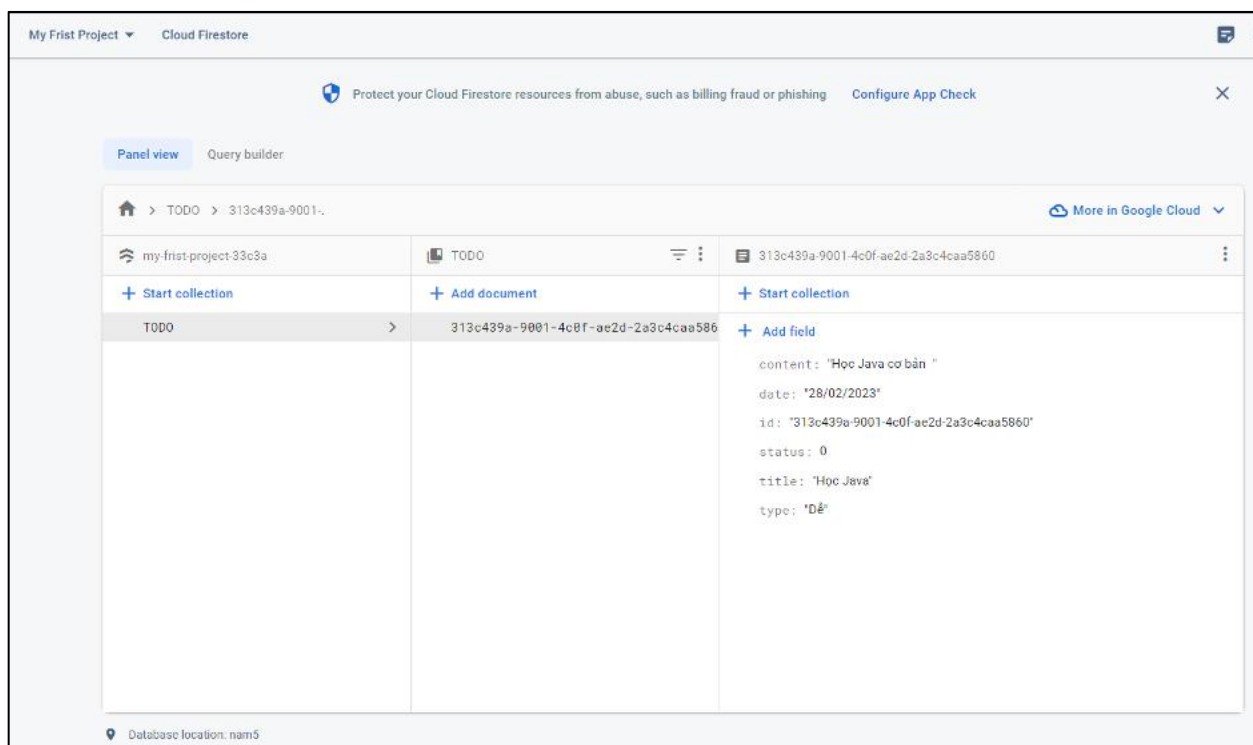
Thay đổi đoạn code thêm một công việc mới, thay vì lưu trữ dữ liệu vào SQLite, ta tiến hành tạo một collection mới trên **FirebaseFirestore** và lưu trữ dữ liệu

```
//Lấy dữ liệu từ các ô nhập  
String title = edit_title.getText().toString();  
String content = edit_content.getText().toString();  
String date = edit_date.getText().toString();  
String type = edit_type.getText().toString();  
  
// id ở đây ta sẽ sử dụng UUID để tạo ra các mã ngẫu nhiên  
// Ta cũng sẽ sử dụng mã id này để đặt tên cho Document để tiện thao tác dữ liệu  
String id = UUID.randomUUID().toString();  
ToDo toDo = new ToDo(id, title, content, date, type, status: 0);  
HashMap<String, Object> mapToDo = toDo.convertHashMap();  
  
database.collection(TODO).document(id) // Dùng id để đặt tên cho document  
    .set(mapToDo) // Đưa dữ liệu vào database  
    .addOnSuccessListener(new OnSuccessListener<Void>() {  
        @Override  
        public void onSuccess(Void unused) {  
            Toast.makeText(context: MainActivity.this, text: "Thêm công việc thành công", Toast.LENGTH_SHORT).show();  
        }  
    }).addOnFailureListener(new OnFailureListener() {  
        @Override  
        public void onFailure(@NonNull Exception e) {  
            Toast.makeText(context: MainActivity.this, text: "Thất bại", Toast.LENGTH_SHORT).show();  
        }  
    });
```

Chạy lại project và thực hiện thêm một công việc mới



Sau khi thêm thành công, ta tiến hành kiểm tra **Firestore** trên Firebase, ta được kết quả như hình



BÀI 3: Thực hiện chức năng lắng nghe khi dữ liệu thay đổi trên Firestore

Tạo một hàm mới tên **ListenFirestore**, trong hàm này sẽ lắng nghe sự thay đổi dữ liệu trên Firestore (Thêm, Sửa, Xóa)

```
private void ListenFirestore() {
    database.collection(TODO).addSnapshotListener(new EventListener<QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {
            if (error != null) {
                Log.e(tag: "TAG", msg: "Listen failed", error);
                return;
            }

            if (value != null) {
                for (DocumentChange dc : value.getDocumentChanges()) {
                    switch (dc.getType()) {
                        case ADDED: //Sự kiện khi có 1 document được thêm vào
                            dc.getDocument().toObject(Todo.class);
                            listToDo.add(dc.getDocument().toObject(Todo.class));
                            listToDoAdapter.notifyItemInserted(position: listToDo.size() - 1);
                            break;
                        case MODIFIED: //Sự kiện khi có 1 document được cập nhật
                            Todo updateTodo = dc.getDocument().toObject(Todo.class);
                            //Nếu vị trí của đối tượng tương đương với vị trí mới
                            if (dc.getOldIndex() == dc.getNewIndex()) {
                                //Set thay đổi đối tượng chưa cập nhật (cũ) thành đối tượng đã cập nhật
                                listToDo.set(dc.getOldIndex(), updateTodo);
                                //Thông báo cho adapter có 1 đối tượng đã cập nhật
                                listToDoAdapter.notifyItemChanged(dc.getOldIndex());
                            } else {
                                //Nếu khác vị trí, sẽ xóa đối tượng ở danh sách
                                listToDo.remove(dc.getOldIndex());
                                //Và thêm lại
                                listToDo.add(updateTodo);
                                listToDoAdapter.notifyItemMoved(dc.getOldIndex(), dc.getNewIndex());
                            }
                        case REMOVED: //Sự kiện khi có 1 document bị xóa khỏi collection
                            dc.getDocument().toObject(Todo.class);
                            listToDo.remove(dc.getOldIndex());
                            listToDoAdapter.notifyItemRemoved(dc.getOldIndex());
                            break;
                    }
                }
                //Trường hợp xảy ra các điều kiện trên là do bản thân của firebase
                // sẽ sắp xếp lại dữ liệu cho chúng ta (nếu chúng ta ko đặc điều kiện sắp xếp cụ thể khi tr
                break;
            }
        }
    });
}
```

Gọi hàm **ListenFirestore()** trong **onCreate()**

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    //Kết nối với database hiện tại
    database = FirebaseFirestore.getInstance();
    ListenFirestore();
    getData();
}
```

Kể từ bây giờ, khi bạn thực hiện thay đổi dữ liệu trên **Firestore**, dữ liệu bên dưới ứng dụng sẽ tự động cập nhật

BÀI 4: Thực hiện chức năng cập nhật và xóa một công việc

Trong dialog cập nhật thông tin công việc trong Adapter, thay đổi hàm updateToDo bằng đoạn code sau:

```
database.collection(TODO).document(toDo.getId()) //gán ID Document cần update
    .update(toDo.convertHashMap())
    .addOnSuccessListener(new OnSuccessListener() {
        @Override
        public void onSuccess(Object o) {
            Toast.makeText(context: MainActivity.this, text: "Cập nhật thành công", Toast.LENGTH_SHORT).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Log.e(tag: "Lỗi", msg: "onFailure: " + e);
        }
    });
```

Tương tự với hàm xử lý xóa một công việc trong Adapter

```
database.collection(TODO).document(id) //truyền id công việc cần xóa
    .delete().addOnSuccessListener(new OnSuccessListener() {
        @Override
        public void onSuccess(Object o) {
            Toast.makeText(context: MainActivity.this, text: "Xóa thành công", Toast.LENGTH_SHORT).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Log.e(tag: "Lỗi", msg: "onFailure: " + e);
        }
    });
```

Cập nhật trạng thái công việc:

```
int value = check ? 1 : 0;
database.collection(TODO).document(id)
    .update(field: "status", value).addOnSuccessListener(new OnSuccessListener() {
        @Override
        public void onSuccess(Object o) {
            Toast.makeText(context: MainActivity.this, text: "Cập nhật trạng thái thành công", Toast.LENGTH_SHORT).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) { Log.e(tag: "Lỗi", msg: "onFailure: " + e); }
    });
```

BÀI 5: GV CHO THÊM

***** YÊU CẦU NỘP BÀI:**

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết ---